

REMARKS

Claim 4 has been amended to correct an informality. Accordingly, claims 1-82 are presented for examination.

The specification has been amended to remove the browser-executable code objected to by the examiner.

Selected of the claims have been rejected under 35 USC 102 over Jueneman ("Message Authentication," Proceedings of the 1983 IEEE Symposium on Security and Privacy, hereafter Jueneman), or under 35 USC 103 over Jueneman in view of Jakubowski (US Patent 6,226,742) or under 35 USC 103 over Jueneman in view of Enichen et al. (US Patent 6,333,983), or under 35 USC 103 over Jueneman in view of Enichen and further in view of Jakubowski. These rejections are respectfully traversed and reconsideration thereof is requested.

The reference Jueneman is deficient and does not meet the claim language for at least the following reasons:

1. Jueneman teaches the use of MAC or cryptographic MDC codes, not the claimed language "*applying a non-cryptographic Manipulation Detection Code (MDC) function.*" In fact, Jueneman specifically teaches away from using non-cryptographic MDC. "In effect, we were not successful in generating the redundancy required for authentication using only non-secret quantities." Jueneman at page 53, column 2, 3rd paragraph, lines 1-3 (emphasis added).
2. Jueneman provides no disclosure or suggestion of using a randomization function. Thus, Jueneman does not meet the claim language "*performing a randomization function over said plurality of hidden ciphertext blocks.*" (emphasis added)

These deficiencies are not met by either of Enichen et al. or Jakubowski et al. and one of ordinary skill in the art would not be motivated to modify Jueneman with the teachings of

either of these references. In contrast to Applicant's claimed inventive method, Enichen et al. do not use MDC's and also makes two cryptographic processing passes. See column 12, lines 46-48 "the encrypted data is reencrypted under a weak key W and the result then further encrypted." See also the two passed in claim 1 of Enichen that calls for "(a) an encryption function for encrypting . . . (b) a reencryption function for reencrypting."

Likewise, Jakubowski et al. teach the use of a MAC, not MDC. See the Jakubowski et al. abstract. Note that MAC's inherently are cryptographic and cannot meet the claim language "*applying a non-cryptographic Manipulation Detection Code (MDC) function.*" Additionally, Jakubowski requires two cryptographic processing passes, i.e., a first pass to generate the MAC (see Fig. 4A and column 9, line 44-47), and a second pass that either uses a stream cipher or a backward CBC (see Fig. 4A and column 10, lines 31-42 and column 11, lines 48-55).

In respect to claims 1 and 65, the Office Action asserts that Jueneman discloses all of the claim elements, citing page 37, col. 1, last paragraph – page 38, 2nd paragraph for the "receiving" step, and citing the Abstract, page 40, col.1, 3rd paragraph – page 41, 3rd paragraph for the "*creating an MDC block of ℓ -bits in length that includes the result of applying a non-cryptographic Manipulation Detection Code (MDC) function [. . .] making one and only one processing pass with a single cryptographic primitive over each of said equal-size blocks and the MDC block to create a plurality of hidden ciphertext blocks each of ℓ -bits in length; and performing a randomization function over said plurality of hidden ciphertext blocks each of ℓ -bits in length.*"

Regarding claim 28 the Office Action cites page 40, col.1, 3rd paragraph – page 41, col. 1, 4th of Jueneman and asserts that Jueneman discloses "*Verifying integrity of the plaintext blocks using a non-cryptographic Manipulation Detection Function (MDC) function; outputting the plurality of plaintext blocks as an accurate plaintext string if the integrity verification passes; and outputting a failure indicator if the integrity verification fails*"

The Office Action assertions are incorrect. We show this for all the three citations made by this Office Action. We begin with the limitation “*applying a non-cryptographic Manipulation Detection Code (MDC) function*” and the second text cited, namely Abstract, page 40, col. 1, 3rd paragraph – page 41, 3rd paragraph. We defer the response to the first text cited to the response regarding the same text cited in the rejection of claims 35 and 41 below. In the Abstract, in the paragraphs cited by the Office Action, and throughout the entire published article, Jueneman seeks “[t]echniques for combining secrecy and authentication in only one encryption pass, using a Manipulation Detection Code generated by noncryptographic means” (Abstract, lines 9-13). But by the Jueneman authors’ own admission, Jueneman does not find any such techniques. Jueneman explicitly warns that use of such MDC techniques are almost surely doomed to fail. Referring to the use of non-cryptographic MDC functions for generating the redundancy necessary for validating authenticity of encrypted messages, Jueneman concludes:

“In effect, we were not successful in generating the redundancy required for authentication using only non-secret quantities. In fact, the under-determined knapsack attack of Coppersmith and the delayed transmission OFB attack both indicate that the search for an MDC technique that is both independent of the encryption method and strictly a complicated function of the plaintext itself is almost surely doomed to fail in the case of known plaintext” (Summary and Conclusions, page 53, 2nd column, 3rd paragraph) (emphasis added).

Jueneman explicitly defines a non-cryptographic Manipulation Detection Code (MDC) as a redundancy function that (1) is applicable only to plaintext data (page 35, 1st column 2nd paragraph, lines 6-9, and page 37, 2nd column, 3rd paragraph, lines 1-4), (2) does not have cryptographic properties, and (3) does not require a secret quantity, such as a secret cryptographic key, for its computation. Jueneman states:

“Manipulation Detection Codes were defined to be a class of functions which do not involve a strong cryptographic function in and of themselves, and thus do not require a secret key” (Summary and Conclusions section, page 53, 1st column, 3rd paragraph, emphasis added).

Applicant notes that Jueneman’s definition of a non-cryptographic MDC function is consistent with Applicant’s and with that used by those of ordinary skill in the art. However, in contrast to Jueneman, who teaches that such functions are almost surely doomed to fail in

protecting authenticity (integrity) of encrypted messages in several attacks, Applicant's teaches an inventive method for such protection using Manipulation Detection Codes that (1) are generated by non-cryptographic means, (2) do not require any secret quantities in their computation, and (3) are applicable only to the message plaintext itself, and thus independent of the encryption method. Applicant's claim 1 teaches

"An encryption method for providing both data confidentiality and integrity for a message, comprising the steps of: receiving an input plaintext string comprising a message and padding it as necessary such that its length is a multiple of ℓ -bits; partitioning the input plaintext string a length that is a multiple of ℓ -bits into a plurality of equal-size blocks of ℓ -bits in length; creating an MDC block of ℓ -bits in length that includes the result of applying a non-cryptographic Manipulation Detection Code (MDC) function to the plurality of the equal-size blocks [...]" (emphasis added).

In a direct and unambiguous **teach-away** from Applicant's claim 1, Jueneman concludes that it fails to find such techniques. Throughout the entire published article, Jueneman explicitly rejects the use of non-cryptographic MDC functions known in the art, such as block-wise Exclusive-OR function (modulo 2 addition) and linear addition modulo 2^{64} (Abstract, page 40, first column, 3rd paragraph – page 49, 1st column, 2nd paragraph), for use with standard encryption modes (FIPS PUB 81) to protect confidentiality and authenticity (integrity). Jueneman explicitly teaches that non-cryptographic MDCs fail to preserve integrity of encrypted (ciphertext) messages and therefore must be rejected. Jueneman states in boldfaced letters:

"Thus, the block-wise Exclusive-OR (modulo 2 addition) technique for computing a Manipulation Detection Code (MDC) fails to detect the transposition or pair-wise insertion of spurious blocks of ciphertext when used with Cipher Block Chaining and therefore must be rejected" (page 41, 1st column, 4th paragraph).

This is a direct teach-away to Applicant's claim language (viz. claims 1, 28, 65 and 21 cited below) and claimed inventive method. Jueneman also lists a series of vulnerabilities of the Exclusive-OR MDC to unknown plaintext attacks, which renders the use of MDCs useless in practice (page 43, 1st column, last paragraph entitled "Vulnerabilities of Exclusive-OR MDC to Unknown Plaintext Attacks"). Jueneman further teaches that use of the non-cryptographic MDC defined by linear addition modulo 2^k (e.g., $k=64$) does not preserve the

integrity of encrypted (ciphertext) messages and therefore must also be rejected. Jueneman states:

“In any case, as the next section points out, the linear addition technique is not totally satisfactory from an architectural standpoint, because it does not withstand bit manipulation attacks with Output Feedback mode, and is subject to known plaintext attacks as well” (page 46, 1st column, 4th paragraph, last 4 lines).

In teaching the use of “a Manipulation Detection technique based upon the use of linear addition modulo 2^k ,” Jueneman also states:

“Further, we shall see in the next section that it does not stand up under a number of known plaintext attacks, at least when used with a common key across multiple messages. It therefore does not meet our criteria for the absolute detection of any modification” (page 47, 1st column, 3rd paragraph),

and concludes in boldfaced letters:

“Therefore, on the basis of the desiderata this technique should be rejected, at least on its present form” (page 47, 1st column, 3rd paragraph).

In the Summary and Conclusions section, Jueneman again explicitly rejects the only two kinds of non-cryptographic MDC disclosed stating:

“Analysis has shown that a previously proposed Manipulation Detection Code based on the use of a block-wise Exclusive-OR operation (modulo 2 addition) is not sound, and can be easily compromised by transposition and data insertion attacks.

A linear addition modulo 2^k MDC technique was evaluated and was also recommended for rejection, because it provided insufficient protection for use with Output Feedback mode.” (page 53, 1st column, page 53, last paragraph – 2nd column, first paragraph, lines 1-4, emphasis added).

This is a direct teach-away to Applicant’s claim language and inventive method (viz., claims 1, 28, 65 and 21) which require that the non-cryptographic MDC protect message integrity.

Jueneman does teach the use of a cryptographic MDC, i.e., the QCMDC. By Jueneman’s own definition, the QCMDC is not a non-cryptographic MDC. The fact that a QCMDC is a cryptographic MDC (as opposed to a non-cryptographic one as taught by Applicant’s language of claims 1, 28, and 65) is made clear by Jueneman in at least three

specific and unambiguous ways. First, Jueneman makes it clear that the QCMDC technique “is based on several techniques used in the generation of software hashing techniques and pseudo-random-number generators” (page 49, 2nd column, 7th paragraph). As all those of ordinary skill in the art understand, these techniques are basic cryptographic (as opposed to non-cryptographic) techniques. Thus Applicants’ claim language that requires the use of a non-cryptographic Manipulation Detection Code (MDC) is not satisfied since Jueneman’s QCMDC is cryptographic.

Second, Jueneman requires at least one secret initial value C, and recommends another secret value, namely, seed S, be used for the computation of the QCMDC. Jueneman states, in boldfaced letters, that

“the use of the secret seed S in addition to the secret initial value C is recommended on the basis of general prudence” (page 52, 1st column, 4th paragraph).

The secret initial value C and secret seed S cannot be part of the plaintext since, by definition, the plaintext is not secret since it is clearly known to the party that encrypts the message. As Jueneman makes it clear, C and S cannot be known to this party (viz., page 50, 1st column, 3rd paragraph, lines 4-7, and page 50, 1st column, last paragraph, lines 4-6). However, by Jueneman’s own definition, a non-cryptographic MDC “is computed solely as a function of the plaintext” (page 37, 2nd column, 3rd paragraph, lines 1-4). Hence any non-cryptographic MDC cannot be computed using secret values C and S, as required by the computation of the QCMDC, and thus the QCMDC cannot possibly be a non-cryptographic MDC. Further, in referring to the computation of the secret initial value C of the QCMDC in a way that avoids transmission of C between the message sender and receiver, Jueneman explicitly teaches the use of a strong cryptographic function. That is, Jueneman teaches that C be computed by the encryption of constant zero using the secret key K shared by the sender and receiver, and makes it clear that

“some non-cryptographic permutation or inversion of the bits of K would not be considered safe” (page 52, 1st column, last paragraph).

Similarly, Jueneman teaches that the secret seed S of the QCMDC

“could be purely random, or it could be the encrypted value of a sequence number or date/time (which would appear doubly encrypted in the ciphertext)” (page 52, 2nd column, first four lines).

One of ordinary skill in the art recognizes the indisputable fact that both secret initial value C and secret seed S taught by Jueneman’s QCMDC must be computed by cryptographic means (e.g., random number generation or encryption) and hence the computation of QCMDC is, by definition, cryptographic (as opposed to non-cryptographic). Thus Applicant’s claim language that requires the use of a non-cryptographic Manipulation Detection Code (MDC) is not satisfied since Jueneman’s QCMDC is cryptographic.

Third, Jueneman explicitly admits that the QCMDC needs a secret quantity in its computation and that it has characteristics of cryptographic (as opposed to non-cryptographic) functions. Referring to the QCMDC, Jueneman states:

“the secret quantity must be treated in many respects as a secret cryptographic key. As a consequence the MDC takes on some of the characteristics of a MAC” (page 53, 2nd column, 2nd paragraph, last 4 lines),

where MAC stands for Message Authentication Code, which is understood by Jueneman (viz., page 37, 2nd column, 3rd paragraph, lines 4-7) and those of ordinary skill in the art to be a basic cryptographic function (as opposed to a non-cryptographic one). Thus, by Jueneman’s own explicit admission, the QCMDC cannot possibly be non-cryptographic since it uses the secret quantity that “*must be treated in many respects as a secret cryptographic key.*” Further, also by Jueneman’s own admission, the QCMDC cannot possibly be non-cryptographic since it “*takes on some of the characteristics of a MAC,*” which Jueneman and those of ordinary skill in the art recognize to be a basic cryptographic function.

Thus Applicant’s claim language “*applying a non-cryptographic Manipulation Detection Code (MDC) function*” is not satisfied and Jueneman’s MDC techniques cannot possibly be used in the Applicant’s claimed inventive method. In contrast with Jueneman, the Applicant’s claimed inventive method uses only non-cryptographic MDC functions that protect message authenticity (integrity) when used with Applicant’s claimed inventive encryption method. Since Jueneman teaches away from the use of non-cryptographic MDCs because they do not protect message authenticity (integrity) and teaches the use of

cryptographic MDCs, none of Jueneman's teachings meet Applicant's claimed inventive method.

Regarding dependent claim 21, this claim explicitly requires the use of the Exclusive-OR (addition modulo 2) MDC (*"The method as defined in claim 1, wherein said non-cryptographic MDC function is a bit-wise exclusive-or function."*), which Jueneman explicitly rejects.

The claim 1 and 65 element *"performing a randomization function over said plurality of hidden ciphertext blocks"* is not met by Jueneman. Applicant's claimed inventive encryption method (viz., claim 1 and 65 language) requires that the ciphertext blocks be generated by the single processing pass with a single cryptographic primitive (*"making one and only one processing pass with a single cryptographic primitive"*) over each of the plaintext blocks of a message and the MDC block (called the "hidden ciphertext blocks") be randomized to create the ciphertext message that is output (*"performing a randomization function over said plurality of hidden ciphertext blocks to create a plurality of output ciphertext blocks"*). We quote from claims 1 and 65:

"An encryption method for providing both data confidentiality and integrity for a message, comprising the steps of: [...] making one and only one processing pass with a single cryptographic primitive over each of said equal-size blocks and the MDC block to create a plurality of hidden ciphertext blocks each of ℓ -bits in length; and performing a randomization function over said plurality of hidden ciphertext blocks to create a plurality of output ciphertext blocks each of ℓ -bits in length."

None of Jueneman standard encryption modes includes either explicitly or implicitly the inventive output-randomization step of the Applicant's claimed method (emphasized in the above quotation) and, without this inventive step, Jueneman's techniques could not possibly be used by the Applicant to meet the claim language and objectives.

Referring to claim 28, Jueneman does not meet either of the limitations

*"performing a reverse **randomization function** on each of the selected $n+1$ ciphertext blocks"* or

“outputting the plurality of plaintext blocks as an accurate plaintext string if the integrity verification passes; and outputting a failure indicator if the integrity verification fails.” (emphasis added)

For the first limitation, as noted above there is no suggestion of randomization in Jueneman. Similarly, there is no suggestion of reverse randomization in Jueneman.

For the second limitation, the Office Action cites page 40, col. 1, 3rd paragraph – page 41, col. 1, 4th. In this cited text, Jueneman discloses a test procedure for integrity verification that outputs inaccurate plaintext strings when it passes. Referring to the inaccuracy of the test procedure that verifies the integrity of plaintext blocks in ciphertext decryption, Jueneman denotes the plaintext blocks by X_i and the ciphertext blocks by Y_i , and states:

“To show that this procedure does not detect certain data manipulation, X_i is expressed in terms of Y_i according to Equation 3 to provide the expression for the sum, S , of all X as follows:

[...Equation 6...]

Since the terms in Equation 6 can be summed modulo 2 in any order, it follows at once that the MDC would not change if the ciphertext blocks were permuted. Thus the proposed modulo 2 addition or bit-wise Exclusive-OR technique for computing the MDC would not detect various kinds of rearrangements of the text occurring on 64 bit boundaries. Furthermore, if additional data, Y^* , is inserted such that

[... Equation 7...]

then such an insertion can not be detected either. The simplest way of satisfying Equation 7 is by inserting the spurious data Y^* into two different places in the message (on 64-bit boundaries, and prior to the last block).” (emphasis added)

In contrast, Applicant’s claimed inventive method teaches an integrity verification procedure used at ciphertext decryption that outputs an accurate plaintext string when verification passes. Applicant’s claim 28 teaches:

“A decryption method that is the inverse of an encryption method which provides both data confidentiality and integrity, comprising the steps of: [...] verifying the integrity of the plaintext blocks using a non-cryptographic Manipulation Detection Code (MDC) function; outputting the plurality of plaintext blocks as an accurate plaintext string if the integrity verification passes; and outputting a failure indicator if the integrity verification fails.” (emphasis added)

Since Jueneman's test procedure fails to detect integrity violations by certain data manipulation, it outputs an inaccurate plaintext string when ciphertext data is manipulated. This is explicitly ruled out by the Applicant's claimed inventive method and cannot be used to satisfy Applicant's claim language and objectives.

The other claims that depend from claim 1 and the other independent claims rejected under 35 USC 102 are allowable for the reasons cited above, and also in view of the respective limitations recited thereon. It is unnecessary to argue these additional limitations in view of the deficiencies of the cited references.

Claim 3 and various other claims are rejected under 35 USC 103 over Jueneman in view of Jakubowski, with Jueneman being cited for disclosing *"applying the non-cryptographic MDC function to the partitioned plaintext blocks; and combining the result with a secret, ℓ -bit random vector generated on a per-message basis to obtain said MDC block (see page 40, 3rd paragraph-page 41, col. 1, 4th paragraph)."* However, Jueneman does not generate any *"secret, ℓ -bit random vector generated on a per-message."* Thus, it would be impossible *"to obtain said MDC block"* in Jueneman. Moreover, the Jueneman text cited in this Office Action does not show the combination of the MDC result with anything before encryption. Hence, a combination using a bit-wise exclusive-or operation as required by Applicant's claim 4 would also be impossible. Claim 5 and various other claims are rejected under 35 USC 103 over Jueneman in view of Jakubowski, with Jakubowski being cited for disclosing *"comprising the step of generating said secret random vector from a secret random number generated on a per-message basis.(see Jakubowski, col. 9, lines 4-58)."* However, Jakubowski does not generate any *"secret random vector from a secret random number generated on a per-message basis,"* and as noted previously, Jakubowski does not disclose the use of MDC's. Hence it would also be impossible *"to obtain said MDC block"* in Jakubowski. As noted previously, Jakubowski also would not suggest anything to one of ordinary skill in the art with respect to Jueneman because Jakubowski does not disclose the use of MDC's and uses two cryptographic passes. Thus Jueneman and Jakubowski's disclosure cited in this Office Action could not possibly satisfy Applicant's claim language.

As noted for the previous rejection, various dependent claims add further limitations and distinguish over the art also for these additional limitations. It is unnecessary to argue these limitations in view of the deficiencies in the cited art.

Selected claims including parallel encryption claim 35 and parallel decryption claim 41 were rejected under 35 USC 103(a) over the reference combination of Jueneman in view of Enichen. Selected elements of claims 35 and 41 are recited below:

“A method for parallel encryption processing of a message comprising the steps of [...]:

“concurrently presenting each different one of said plurality of input plaintext segments to a different one of a plurality of encryption processors, each of said different processors using a different ℓ -bit secret random number per segment to obtain a ciphertext segment using an encryption method providing both data confidentiality and integrity with a single processing pass over the input plaintext segment and a single cryptographic primitive, and using a non-cryptographic Manipulation Detection Code function, wherein said single cryptographic primitive is a ℓ -bit block cipher using a secret first key.” (emphasis added)

“A method for parallel decryption processing of a message comprising the steps of:

“obtaining a different secret random number per ciphertext segment from a secret random number in the same manner as at a parallel encryption method;
decrypting each ciphertext segment using said different secret random number per ciphertext segment to obtain a plaintext segment, using a decryption method that is the inverse of an encryption method used in the parallel encryption method that provides both data confidentiality and integrity with a single processing pass over the input plaintext segment and a single cryptographic primitive, wherein said single cryptographic primitive is a ℓ -bit block cipher using a secret first key, and using a non-cryptographic Manipulation Detection Code function for verifying integrity of the plaintext blocks of each plaintext segment”. (emphasis added)

It has been previously shown that Jueneman does not disclose or suggest using a non-cryptographic MDC. Likewise, Jueneman does not disclose or suggest using a secret random number for encrypting or decrypting.

Regarding Jueneman, the first text cited in this Office Action (i.e., page 37, col. 1, last paragraph – page 38, 2nd paragraph entitled “Message Authentication Codes,” which is also the first text cited in regard to claims 1 and 65 above), refers to the use of MAC’s. This is explicitly ruled out by the Applicant’s claim language because use of MAC’s in maintaining

secrecy and authenticity (integrity) would require two cryptographic processing passes over the message blocks, namely one pass for computing the MAC and the other for message encryption. The fact that two cryptographic processing passes are required when MAC's are used to protect the authenticity (integrity) of encrypted messages is explicitly stated by Jueneman and known to those of ordinary skill in the art. Jueneman states (in boldfaced letters):

“As far as is known, if two keys are used, one for encipherment for secrecy and one for the generation of the Message Authentication Code (MAC), then a strong procedure is obtained and the order of operations is unimportant.” (page 38, 1st column, 2nd paragraph, emphasis added).

This section of Jueneman cited in this Office Action is a direct teach-away to Applicant's claim language.

In the second text cited in this Office Action regarding Jueneman, namely page 40, col. 1, 3rd paragraph – page 41, col. 1, 4th paragraph – Section titled Manipulation Detection Codes), Jueneman teaches only that the use of the Exclusive-OR (modulo 2 addition) fails to detect simple integrity attacks and should be rejected. Jueneman concludes the section cited in this Office Action, in boldfaced letters:

“Thus, the block-wise Exclusive-OR (modulo 2 addition) technique for computing a Manipulation Detection Code (MDC) fails to detect the transposition or pair-wise insertion of spurious blocks of ciphertext when used with Cipher Block Chaining and therefore must be rejected” (page 41, 1st column, 4th paragraph).

This is a direct teach-away to Applicant's claim language and inventive method. In contrast with Jueneman's teaching above, Applicant's claimed inventive method in claim 21 teaches explicitly the use the Exclusive-OR (modulo 2 addition) MDC, which Jueneman explicitly rejects, with the Applicant's inventive encryption method to preserve the integrity of encrypted messages (i.e., of ciphertext messages). Applicant's claim 21 explicitly states:

“The method as defined in claim 1, wherein said non-cryptographic MDC function is a bit-wise exclusive-or function.” (emphasis added)

In referring to Enichen, the Office Action states that “Jueneman does not disclose using parallel processing to implement the encryption and decryption process” and asserts that Enichen does, citing (Enichen, col. 2, lines 40-60). We note that this text is in the Description of the Preferred Embodiment section of Enichen, is the one and only instance in the entire text of Enichen where the term “parallel” appears, and this term clearly and unambiguously refers to the IBM Parallel Enterprise Server G3 or G4 processor and not to the CF (which comprises the cryptographic coprocessor) or to the CFAP, where Enichen performs message encryption and decryption. Enichen specifies that all encryption and decryption operations are performed by CFAP or CF, col. 2, lines 61-67 – col. 3, lines 1-5.

“More particularly, while the present invention is not limited to a specific platform, in an IBM S/390 environment CF 102 may comprise the cryptographic coprocessor of an IBM Parallel Enterprise Server G3 or G4 processor and CFAP 106 may comprise the Integrated Cryptographic Service Facility (ICSF), a component of the OS/390 operating system.” (col. 2, lines 50-56). [nb. CF stands for Cryptographic Facility and CFAP for Cryptographic Facility Access Program].

Further, Enichen explicitly teaches sequential encryption (and not parallel encryption, as incorrectly asserted by the Office Action) where each plaintext is partitioned only into 64-bit (8-byte) blocks and processes them one at a time. Enichen states:

“CFAP 106 then processes the data X1-Xn, one 64-bit block Xi at a time, using the procedure shown in FIG. 7, to obtain corresponding encrypted data blocks Yi (step 606), which it returns to application 110 (Step 608).” (col. 8, lines 46-49, emphasis added).

By definition, processing one 64-bit block Xi at a time represents sequential (and not parallel) encryption. Enichen repeatedly shows that the encryption and decryption procedures are sequential, and not parallel. For example,

“On the first invocation, CFAP supplies X1 as a single block input message and the ICV value received from application 110; the return value from CF 102 is saved as Y1. On subsequent invocation, CFAP 106 supplies the corresponding block Xi as a single-block input message and the previous value Yi-1 as an ICV, and saves the output value as Yi.” (col. 9, lines 6-12, of Section titled “Encryption Procedure,” emphasis added.)

Since the encryption of input block X_i requires the output block Y_{i-1} as above, message encryption can be only sequential (and not parallel). Further, in referring to the decryption procedure, Enichen states:

“CFAP 106 then processes the DES-encrypted data one 8-byte block Y_i at a time, beginning with block Y_1 and continuing through block Y_n , using the procedure shown in FIG. 12 (step 1106).” (col. 10, lines 62-65, of Section titled “Decryption Procedure,” emphasis added),

and

“Finally, when all data Y_1 - Y_n has been deciphered, CFAP 106 returns the deciphered data X_1 - X_n to the application 110 (step 1108).” (col. 11, lines 1-3, of Section titled “Decryption Procedure”).

Again, by definition, processing one 8-byte block Y_i at a time at decryption represents sequential (and not parallel) decryption. Other instances exist where Enichen shows that the decryption procedures are sequential and not parallel. For example, in providing details for the decryption procedure shown in FIG. 12, Enichen states:

“On the first pass through the FIG. 12 loop, the chaining vector is the initial chaining vector ICV received from application 110. On each subsequent pass, the chaining vector is the previous eight bytes of the decrypted data. If there are any remaining blocks of data Y_i , the procedure is repeated from step 1202 for each additional block of data, processing 8 bytes at a time, until all such blocks have been processed (step 1210).” (col 11, last line 66 – col. 12, lines 1-8, of Section titled “Decryption Procedure” emphasis added).

Enichen et al. could not be any clearer that both their claimed encryption and decryption procedures are sequential (and not parallel). In further contrast to Enichen, Applicant’s language of claim 41 provides:

“partitioning said ciphertext string into a plurality of ciphertext segments; concurrently presenting said plurality of ciphertext segments to a plurality of processors; obtaining a different secret random number per ciphertext segment from a secret random number in the same manner as at a parallel encryption method; decrypting each ciphertext segment using said different secret random number per ciphertext segment to obtain a plaintext segment, using a decryption method that is the inverse of an encryption method used in the parallel encryption method that provides both data confidentiality and integrity with a single processing pass over the input plaintext segment and a single cryptographic primitive, wherein said single cryptographic primitive is a ℓ -bit block cipher using a secret first key, and using a non-cryptographic Manipulation Detection Code function for verifying integrity of the

plaintext blocks of each plaintext segment; assembling the plurality of plaintext segments into a plaintext string; and verifying the integrity of the plaintext segments and their sequence and outputting the plaintext string if the integrity verification passes.” (emphasis added)

Thus the language of claims 35 and 41 is cannot possibly be satisfied by Enichen’s teaching. Further, Enichen is a direct teach-away to the Applicant’s language of claims 35 and 41 since Enichen teaches sequential (not parallel) message encryption and decryption. Since neither Jueneman (as admitted in the Office Action) nor Enichen teach parallel encryption and decryption, the Office Action reason for rejecting Applicant’s claims 35 and 41 is clearly incorrect.

Further, not only does Enichen not teach parallel encryption and decryption, but also it explicitly teaches two processing passes with a cryptographic primitive over the message data for the encryption operation and two processing passes over the message data for the decryption operation. This is a direct teach-away to the language of Applicant’s claims 35 and 41. For example, Enichen states in describing the decryption operation:

“In the decryption operations described above, the encrypted data is reencrypted under a weak key W and the result then further encrypted under the same weak key W.” (col. 12, lines 46-48, emphasis added).

Further, Enichen’s claim 1 also makes it clear that two processing passes with a cryptographic primitive are required both for the encryption operation and for the decryption operation. Use of two processing passes with a cryptographic primitive is explicitly ruled out by the Applicant’s language in claims 35 and 41 (“*with a single processing pass*”) and inventive method. Thus Enichen is a direct teach-away to Applicant’s claim language.

Since neither Jueneman nor Enichen teach a parallel encryption and decryption method, combining their teaching cannot possibly teach a parallel encryption and decryption method as provided by Applicant’s claims 35 and 41.

As noted for the previous rejection, various dependent claims add further limitations and distinguish over the art also for these additional limitations. It is unnecessary to argue these limitations in view of the deficiencies in the cited art.

The other independent claims rejected under 35 USC 103 are allowable for the reasons cited above, and also in view of the respective limitations recited thereon. It is unnecessary to argue these additional limitations in view of the respective deficiencies of the cited references.

The Commissioner is hereby authorized to charge any additional fees which may be required regarding this application under 37 C.F.R. §§ 1.16-1.17, or credit any overpayment, to Deposit Account No. 19-0741. Should no proper payment be enclosed herewith, as by a check being in the wrong amount, unsigned, post-dated, otherwise improper or informal or even entirely missing, the Commissioner is authorized to charge the unpaid amount to Deposit Account No. 19-0741. If any extensions of time are needed for timely acceptance of papers submitted herewith, Applicant hereby petitions for such extension under 37 C.F.R. § 1.136 and authorizes payment of any such extensions fees to Deposit Account No. 19-0741.

Respectfully submitted,

Date 3/15/05

By 

FOLEY & LARDNER LLP
Washington Harbour
3000 K Street, N.W., Suite 500
Washington, D.C. 20007-5143
Telephone: (202) 672-5485
Facsimile: (202) 672-5399

William T. Ellis
Attorney for Applicant
Registration No. 26,874